Supporting Text Specification:

Card:

The Card class has two attributes, rank and suit that both are integer values. Card also has an equals method that takes other: Card as its parameter and returns a Boolean value. Card has a relationship with both the Deck class and the Slot class.

Deck:

The Deck class has one attribute, cards that is an ArrayList<Card>, an array of cards that compose the deck. The Deck class has three methods, shuffleDeck() that shuffles the cards in the deck and returns a void, drawCard() that will draw a card from the bottom of the deck and returns a Card object, and toString() that prints all the suits and ranks of each card in the deck. The Deck class has a relationship with the Card class.

Slot:

The Slot class has two attributes, type, which gets the type of the card, and cards, which creates a new set of cards for the slot. The Slot class has 8 methods; The createSlot() method that takes a string and creates a slot for the table, the getCards() method that returns cards in a slot, the lookAtTopCard method that returns the top card of a stack, the removeTopCard() method that removes a card from the top of the stack, the addCard(Card) method that adds a card to the top of the stack, the size() method that gets the size of the cards in the slot, the getType() method that gets the type of the slot, and the toString() method that prints the rank and suit names of each card in the slot. The Slot class has a relationship with the Card and SolitaireLogic classes.

SolitaireLogic:

The SolitaireLogic class has no attributes and seven methods. The four methods are setUpFoundationSlots() that sets up the foundation piles (for placing cards), the setUpSlots() method initializes all slots for both the foundation and tableau piles, the setUpTableauSlots() method shuffles and sets up the tableau, the canMoveToFoundation() method that takes a slot from value and slot to value as parameters and determines if the move is valid, the canMoveToTableau() method that takes a slot from value and a slot to value as parameters and determines if the move is valid, the canMove() method that takes a slot from value and a slot to value as parameters and determines if the top card from one slot is allowed to move to the top of another slot, and the moveCard() method that takes a slot from value and a slot to value as parameters and moves a specified card from the top of one slot to the top of another slot. The SolitaireLogic class has a relationship with the Solitaire class.

Solitaire:

The Solitaire class has four attributes, Logic, that defines the logic for the game, GameBoard, that updates the game board or returns card and slot entities, AllSlots that sets the slots by copying the game state history, and GameStateHistory that sets an array list for the game state history. The Solitaire class has 8 methods as follows; undoMove() that reverts to a user’s most recent state, updateBoard() that gets a new instance of the board and updates the slots, getCardGraphics() that gets the graphics for the cards in the slots, resetSelectedCards() that determines if the moving of one card onto another card was successful or not, selectCard() that determines if the card selected is applicable and no card collisions occur, selectSlot() that determines if the slot selected is applicable and no slot collisions occur, and winCondition() that detects whether or not the player has won (the foundation piles much each have 13 cards). The Solitaire class has a relationship with solitaireLogic, Board, and solitaireFrame.

StatsPanel:

The StatsPanel class has 5 attributes, MoveCountText that outputs text indicating the total number of moves made by the user, TimerText that indicates the amount of total time that the user has been playing, Timer that sets the timer either on or off or resets the timer to zero, currentTime that stores the amount of time that the user has been playing, and MoveCount that records the number of moves made by the user. The StatsPanel has 7 methods; setTimer() that turns the timer on or off, reset() that resets the time and moves back to zero, getCurrentTime() that passes how long the user has been playing to the result screen, getMoveCount() that passes how many moves the player made to the result screen, incrementMoveCount() that increases the number of moves the user made by one, decrementMoveCount() that decreases the number of moves made by the user by 2, and actionPerformed() that updates how long the user has been playing. The StatsPanel has a relationship with the SolitaireFrame class.

SolitaireFrame:

The SolitaireFrame class has 8 attributes; Frame\_height that sets the height of the frame, Frame\_Width that sets the width of the frame, Canvas that sets or updates the canvas to hold the graphics, GameMenu that generates a drop down menu that includes a new game, undo move, and quit option, StatusBar that sets the status bar to begin tracking time and the number of moves made, Cards that gets the card graphics, Slots that gets the slots graphics for the Foundation and Tableau piles, and Game that updates the board for a new game. The SolitaireFrame class has 4 methods; start() that creates a new game and grabs cardGraphics to draw to screen and starts the status bar to begin tracking time and number of moves made, undoMove() that will undo the user’s previously made move and updates the move count, exit() that exits the application, and updateScreen() that gets an updated state of the game. SolitaireFrame has a relationship with statsPanel, GameMenu, Solitaire, and DrawPanel.

DrawPanel:

The DrawPanel class has no attributes and 7 methods. mousePressed detects if the mouse has been clicked and if the move is allowed, paint() calls methods to generate graphics for the slots, cards, and win screen, loadImage() helper method to load an image file, drawBackGround() that loads the background image into the frame, drawWinScreen() that outputs a message indicating that the user has won, drawCards() that loads images for the cards and sets their size and location, and drawSlots() that loads the outline for the slots and sets their size and location. The DrawPanel class has a relationship with the SolitaireFrame class.

GameMenu:

The GameMenu class has four attributes; Menu that generates a drop down menu that includes new game, undo move, and quit options, StartMenuItem that creates a new game, UndoMenuItem that reverts the user’s previous move, and ExitMenuItem that exits the application. The GameMenu class has one method; actionPerformed() that detects if either the new game, undo move, or quit options were selected and acts accordingly. The GameMenu class has a relationship with the SolitaireFrame class.

Board:

The Board class has four attributes; CardEntities that gets and generates the card placements within a pile, SlotEntities that gets and generates slot placements, SelectedCardGraphics that selects the card’s graphics, and SlotStartPosition that determines the starting location for the slots. The Board class also has 8 methods; getSelectedCards() that gets the card selected by the user, resetSelectedCards() that clears the selected card so that there is no longer anything selected, checkForCardCollisions() that checks for collisions that may occur between cards, checkForSlotCollision() that checks for collisions that may occur between slots, generateSlotEntities() that determines the location of slots and gives each slot a corresponding id, generateCardEntities() that determines the location of the cards and gives each card an identifier, getSlotEntities() that gets the slot entities for view to initialize its array of slot card graphics, and getCardEntities() that gets the card entities for view to initialize its array of card graphics. The Board class has a relationship with the Solitaire class and the CardGraphic class.

CardGraphic:

The CardGraphic class has three attributes; PosX that gets the X position, PosY that gets the Y position, and Slot\_id that gets the ID of the slot that the card is in. The CardGraphic class also has 6 methods; getCard() gets the card that the graphic is associated with, contains() that checks to see if the location (x,y) is inside of this card graphic, setHighLighted() that sets the card to be highlighted, isHighLighted that is a Boolean for if the card is highlighted or not, getSlotId() that gets the ID of the slot, and toString() that returns the location on the board of the card and what slot it is currently stored in. The CardGraphic class has a relationship only with the Board Class.